IN THE CLAIMS:

The following listing of claims will replace all prior versions, and listings, of claims in the application.

1. (Currently Amended) A method for run-time prediction of a next caller of a shared functional unit, wherein the shared functional unit is operable to be called by two or more callers out of a plurality of callers, the method comprising:

storing a caller history of the shared functional unit;

detecting a calling pattern of the plurality of callers of the shared functional unit, wherein said detecting comprises:

dividing the caller history into a first portion of the caller history and a second portion of the caller history, wherein the first portion and the second portion hold substantially the same portion of the caller history; and

comparing the callers of the first portion of the caller history to the callers of the second portion of the caller history;

predicting the next caller out of the plurality of callers of the shared functional unit; and

loading state information associated with the next caller out of the plurality of callers;

wherein the shared functional unit and the plurality of callers are operable to execute in parallel on a parallel execution unit.

2. (Original) The method of claim 1, wherein the run-time prediction is performed for an application described by a dataflow graph.

3. (Original) The method of claim 1, wherein the run-time prediction is performed for an application programmed in a dataflow language.

4. Cancelled.

5. (Currently Amended) The method of claim 1 [[4]],

wherein said storing the caller history uses a history register, wherein the history register is operable to be divided into two substantially equal parts.

6. (Original) The method of claim 5,

wherein said comparing operates to compare callers in the first part of the history register to the callers in the second part of the history register.

7. (Original) The method of claim 5,

wherein each of the plurality of callers has a unique identification, wherein the unique identification is operable to be used in the caller history.

8. (Original) The method of claim 7,

wherein the history register is operable to store the unique identification of each of the two or more callers calling the shared functional unit by operating analogous to a shift register.

9. (Original) The method of claim 7,

wherein said comparing the callers comprises comparing the unique identifications of the callers in the first portion of the caller history to the unique identifications of the callers in the second portion of the caller history.

10. (Currently Amended) The method of claim 1, [[4:]]

wherein said comparing callers in the first part of the caller history to the second part of the caller history operates to select a periodic portion of the caller history.

11. (Currently Amended) The method of claim 10, further comprising:

using a multiplexer to predict the next caller of the shared functional unit after selecting the periodic portion of the caller history.

12. (Currently Amended) The method of claim 1,

wherein the parallel execution unit comprises one or more of:

an FPGA;

a programmable hardware element;

a reconfigurable logic unit;

a nonconfigurable hardware element;

an ASIC;

a computer comprising a plurality of processors; ~~and~~ or

any other computing device capable of executing multiple threads in parallel.

13. (Currently Amended) The method of claim 1,

wherein the state information comprises one or more of:

execution state;

values of any variable;

previous inputs;

previous outputs; ~~and~~ or

any other information related to execution of a node in a dataflow diagram.

14. (Original) The method of claim 1,

wherein the run-time prediction operates to optimize execution of the nodes in the dataflow program.

15. (Original) The method of claim 1,

wherein the shared functional unit and the plurality of callers are generated from a dataflow program.

16. (Currently Amended) A method for run-time call prediction for resolving resource contention between two or more callers of a shared node in a dataflow program, the method comprising:

storing a caller history of the shared node;

detecting a calling pattern by a plurality of callers of the shared functional unit, wherein said detecting comprises:

dividing the caller history into a first portion of the caller history and a second portion of the caller history, wherein the first portion and the second portion hold substantially the same portion of the caller history; and

comparing the callers of the first portion of the caller history to the callers of the second portion of the caller history;

predicting a next caller out of the plurality of callers of the shared functional unit; and

loading state information associated with the next caller out of the plurality of callers;

wherein the shared functional unit and the plurality of callers are operable to execute in parallel on a parallel execution unit.

17. (Original) The method of claim 16,

wherein the dataflow program comprises of a plurality of nodes, wherein one or more of the plurality of nodes are operable to be called by two or more nodes of the plurality of nodes.

18. (Original) The method of claim 16,

wherein the run-time call prediction operates to optimize execution of the nodes in the dataflow program.

19. (Currently Amended) The method of claim 16,

wherein the dataflow program executes on a parallel execution unit, wherein the parallel execution unit comprises one or more of:

an FPGA;

a programmable hardware element[[s]];

a reconfigurable logic unit;

a nonconfigurable hardware element;

an ASIC;

a computer comprising a plurality of processors; ~~and~~ or

any other computing device capable of executing multiple threads in parallel.

20. Cancelled.

21. (Currently Amended) The method of claim 16 [[20]],

wherein each of the plurality of callers has a unique identification, wherein the unique identification is operable to be used in the caller history.

22. (Original) The method of claim 21,

wherein the history register is operable to store the unique identification of each of the two or more callers calling the shared functional unit by operating analogous to a shift register.

23. (Original) The method of claim 21,

wherein said comparing the callers comprises comparing the unique identifications of the callers in the first portion of the caller history to the unique identifications of the callers in the second portion of the caller history.

24. (Currently Amended) The method of claim 16 [[20]],

wherein said comparing callers in the first part of the caller history to the second part of the caller history operates to select a periodic portion of the caller history.

25. (Currently Amended) The method of claim 16,[[:]]

wherein the shared functional unit and the plurality of callers are generated from the dataflow program.

26. (Currently Amended) A memory medium comprising instructions to generate a program to perform run-time call prediction of a next caller of a shared functional unit, wherein the program is intended for deployment on a parallel execution unit, wherein the program is executable to:

store a caller history of the shared node;

detect a calling pattern of a plurality of callers of the shared functional unit, wherein the shared functional unit is operable to be called by two or more callers out of the plurality of callers, wherein said detecting comprises:

dividing the caller history into a first portion of the caller history and a second portion of the caller history, wherein the first portion and the second portion hold substantially the same portion of the caller history; and

comparing the callers of the first portion of the caller history to the callers of the second portion of the caller history;

predict the next caller out of the plurality of callers of the shared functional unit; and

load state information associated with the next caller out of the plurality of callers;

wherein the shared functional unit and the plurality of callers are operable to execute in parallel on the parallel execution unit.

27. (Previously Presented) The memory medium of claim 26,

wherein the run-time call prediction operates to optimize execution of the nodes in the program.

28. (Currently Amended) The memory medium of claim 26,

wherein the program executes on a parallel execution unit, wherein the parallel execution unit comprises one or more of:

an FPGA;

a programmable hardware element[[s]];

a reconfigurable logic unit;

a nonconfigurable hardware element;

an ASIC;

a computer comprising a plurality of processors; ~~and~~ or

any other computing device capable of executing multiple threads in parallel.

29. Cancelled.

30. (Currently Amended) The memory medium of claim 26 [[29]],

wherein each of the plurality of callers has a unique identification, wherein the unique identification is operable to be used in the caller history.

31. (Original) The memory medium of claim 30,

wherein the history register is operable to store the unique identification of each of the two or more callers calling the shared functional unit by operating analogous to a shift register.

32. (Original) The memory medium of claim 30,

wherein said comparing the callers comprises comparing the unique identifications of the callers in the first portion of the caller history to the unique identifications of the callers in the second portion of the caller history.

33. (Currently Amended) The memory medium of claim 26, [[29:]]

wherein said comparing callers in the first part of the caller history to the second part of the caller history operates to select a periodic portion of the caller history.

34. (Currently Amended) The memory medium of claim 26,

wherein the program comprises one or more of:

program instructions;

digital logic; ~~and~~ or

any type of hardware description used to configure the parallel execution unit.

35. (Previously Presented) The memory medium of claim 26,

wherein the shared functional unit and the plurality of callers are generated from the program.

36. (Original) The memory medium of claim 26,

wherein the program comprises a control and arbitration logic unit that is operable to said detect, said predict, and said load.

37. (Currently Amended) A system for run-time optimization of a dataflow program, the system comprising:

a parallel execution unit;

a plurality of callers;

a shared functional unit, wherein the shared functional unit is operable to be called by two or more callers out of the plurality of callers, wherein the shared functional unit and the plurality of callers are operable to execute in parallel on the parallel execution unit;

an optimization algorithm, wherein the optimization algorithm is operable to:

storing a caller history of the shared node;

detect a calling pattern of the plurality of callers of the shared functional unit, wherein said detecting comprises:

dividing the caller history into a first portion of the caller history and a second portion of the caller history, wherein the first portion and the second portion hold substantially the same portion of the call history; and

comparing the callers of the first portion of the caller history to the callers of the second portion of the caller history;

predict the next caller out of the plurality of callers of the shared functional unit; and

allocate state information associated with the next caller out of the plurality of callers.

38. (Currently Amended) The system of claim 37,

wherein the parallel execution unit comprises one or more of:

an FPGA;

a programmable hardware element[[s]];

a reconfigurable logic unit;

a nonconfigurable hardware element;

an ASIC;

a computer comprising a plurality of processors; ~~and~~ or

any other computing device capable of executing multiple threads in parallel.

39. Cancelled.

40. (Currently Amended) The system of claim 37 [[39]],

wherein each of the plurality of callers has a unique identification, wherein the unique identification is operable to be used in the caller history.

41. (Currently Amended) The system of claim 37 [[39]],

wherein the history register is operable to store the unique identification of each of the two or more callers calling the shared functional unit by operating analogous to a shift register.

42. (Currently Amended) The system of claim 37 [[39]],

wherein said comparing the callers comprises comparing the unique identifications of the callers in the first portion of the caller history to the unique identifications of the callers in the second portion of the caller history.

43. (Currently Amended) The system of claim 37, [[39:]]

wherein said comparing callers in the first part of the caller history to the second part of the caller history operates to select a periodic portion of the caller history.

44. (Original) The system of claim 37,

wherein the shared functional unit and the plurality of callers are generated from the dataflow program.

45. (Original) The system of claim 37,

wherein the optimization algorithm is comprised on a control and arbitration logic unit that is operable to said detect, said predict, and said load.